Package 'fat2Lpoly'

December 18, 2019

Type Package

Title Two-Locus Family-Based Association Test with Polytomous Outcome

Version 1.2.4

Date 2019-12-17

Author Alexandre BUREAU <alexandre.bureau@msp.ulaval.ca> and Jordie Croteau <jorcroteau@gmail.com>

Maintainer Alexandre BUREAU <alexandre.bureau@msp.ulaval.ca>

Depends R (>= 3.2)

Imports kinship2, multgee

Description Performs family-based association tests with a polytomous outcome under 2-locus and 1-locus models
defined by some design matrix.

License GPL

LazyLoad yes

URL https://www.cervo.ulaval.ca/pages_perso_chercheurs/bureau_a/

NeedsCompilation no

Repository CRAN

Date/Publication 2019-12-18 16:20:10 UTC

R topics documented:

design.1locus 2
design.dichotomous 3
design.endo2disease 4
design.full 5
fat2Lpoly 7
fat2Lpoly.allSNPs 11
fat2Lpoly.withinR 12
get.scores.pvalues 14
ped.x.all 16
read.merlin.files 16

Index 19

`design.1locus` *Setting-up design matrices for a polytomous model with a single biallelic marker.*

Description

This function sets up two identical lists of three design matrices, one for each linear predictor of the logit of the three outcome levels defined by the combination of two dichotomous traits against the reference level (0,0) under a model with the main effect of a single biallelic marker.

Usage

```
design.1locus(x, par.constrained, constraints)
```

Arguments

`x`	A numeric vector of values representing genotypes of a biallelic marker. The two homozygous genotypes must be coded 0 and 1, and the heterozygous genotype value depends on the genetic model: 0 (recessive), 1/2 (allelic) or 1 (dominant).
`par.constrained`	Optional matrix of dimensions (`n.levels`-1) x nc specifying the parameter (1, 2 or 3) in the linear predictor for each level involved in the nc constraints BETWEEN the logistic models for different levels of the response variable, one constraint per column. This functionality is not yet implemented.
`constraints`	Optional matrix of dimensions (`n.levels`-1) x nc specifying the nc linear constraints BETWEEN the logistic models for different levels of the response variable, involving the parameters specified in `par.constrained`, one constraint per column. A 0 means that the corresponding parameter is not involved in the constraint. This functionality is not yet implemented.

Details

Let Y_1 and Y_2 be binary variables coding the presence (1) or absence (0) of the two traits (e.g. and endophenotype and a disease trait, respectively). The linear predictors (without intercept) of the logistic functions between outcome levels and the reference level $Y_1 = 0$ and $Y_2 = 0$ are as follows:

$Y_1 = 1, Y_2 = 0 : \beta_{11} X$

$Y_1 = 0, Y_2 = 1 : \beta_{21} X$

$Y_1 = 1, Y_2 = 1 : \beta_{31} X$

The vector X constitute the design matrix for each linear predictor of the above model.

Value

`x.e`	List of 3 design matrices containing the vector X
`x.loc.e`	list of character strings containing the indices of the variables in `x` involved in each term of the model, i.e. "1"
`x.l`	identical to `x.e`
`x.loc.l`	identical to `x.l`

Author(s)

Alexandre Bureau <alexandre.bureau@msp.ulaval.ca>

See Also

`fat2Lpoly`,`design.full`,`design.endo2disease`

`design.dichotomous` *Setting-up the design matrix for a logistic model with two biallelic markers.*

Description

This function sets up two identical lists, each containing a design matrix for the linear predictor of the logit of a dichotomous outcome under a full logistic model with main effects and product terms for two biallelic markers.

Usage

```
design.dichotomous(x,...)
```

Arguments

`x`	A 2-column matrix of numeric values representing genotypes of biallelic markers, with one column per marker and one row per subject. The two homozygous genotypes must be coded 0 and 1, and the heterozygous genotype value depends on the genetic model: 0 (recessive), 1/2 (allelic) or 1 (dominant).
`...`	Additional arguments will be ignored, but must be allowed for compatibility with other design functions.

Details

The linear predictors (without intercept) of the logistic function for $Y = 1$ against the reference level $Y = 0$ has the form:

$\eta_{11}X_1 + \eta_{12}X_2 + \eta_{13}X_1X_2$

The design matrix for the above model is constructed by this function.

Value

x.e	List containing the single design matrix with all terms forming the full 2-locus logistic model
x.loc.e	list of character strings containing the indices of the variables in x involved in each term of the logistic model
x.l	identical to x.e
x.loc.l	identical to x.l

Author(s)

Alexandre Bureau <alexandre.bureau@msp.ulaval.ca>

See Also

fat2Lpoly, design.full

---

design.endo2disease *Setting-up design matrices for the endophenotype-to-disease model.*

Description

This function sets up two lists of three design matrices, one for each linear predictor of the logit of the three outcome levels defined by the combination of two dichotomous traits against the reference level (0,0) under the endophenotype-to-disease model of Bureau et al (2014). The design matrices in the first list contain all terms forming the model, and those in the second list contain all main effect and product terms appearing in the model.

Usage

```
design.endo2disease(x, par.constrained, constraints)
```

Arguments

x	A matrix of dimensions 2 x n of numeric values representing genotypes of biallelic markers, with one column per marker and one row per subject. The two homozygous genotypes must be coded 0 and 1, and the heterozygous genotype value depends on the genetic model: 0 (recessive), 1/2 (allelic) or 1 (dominant).
par.constrained	Optional matrix of dimensions (n.levels-1) x nc specifying the parameter (1 or 2) in the linear predictor for each level involved in the nc constraints BETWEEN the logistic models for different levels of the response variable, one constraint per column. This functionality is not yet implemented.
constraints	Optional matrix of dimensions (n.levels-1) x nc specifying the nc linear constraints BETWEEN the logistic models for different levels of the response variable, involving the parameters specified in par.constrained, one constraint per column. A 0 means that the corresponding parameter is not involved in the constraint. This functionality is not yet implemented.

Details

Let Y_1 and Y_2 be binary variables coding the presence (1) or absence (0) of the endophenotype and the disease trait, respectively. The linear predictors (without intercept) of the logistic functions between outcome levels and the reference level $Y_1 = 0$ and $Y_2 = 0$ specified by the endophenotype-to-disease model are as follows:

$Y_1 = 1, Y_2 = 0 : \beta_{11} X_1 + \beta_e X_1 (1 - X_2)$

$Y_1 = 0, Y_2 = 1 : \beta_{21} X_1$

$Y_1 = 1, Y_2 = 1 : \beta_{31} X_1 + \beta_{33} X_1 X_2$

The design matrices for the above model are constructed by this function.

Value

`x.e`	List of 3 design matrices containing all terms forming the endophenotype-to-disease model
`x.loc.e`	list of character strings containing the indices of the variables in `x` involved in each term of the endophenotype-to-disease model
`x.l`	List of 3 design matrices containing the terms X_1, X_2 and $X_1 X_2$ appearing in the endophenotype-to-disease model.
`x.loc.l`	list of character strings containing the indices of the variables in `x` involved in each term in the list `x.l`

Author(s)

Alexandre Bureau <alexandre.bureau@msp.ulaval.ca>

References

Bureau A., Croteau J., Chagnon, Y.C., Roy, M.-A. and Maziade, M. Extension of the Generalized Disequilibrium Test to polytomous phenotypes and two locus models. Frontiers in Genetics, 5: Article 258.

See Also

`fat2Lpoly`, `design.full`

`design.full` *Setting-up design matrices for a full polytomous model with two biallelic markers.*

Description

This function sets up two identical lists of three design matrices, one for each linear predictor of the logit of the three outcome levels defined by the combination of two dichotomous traits against the reference level (0,0) under a full model with main effects and product terms for two biallelic markers.

Usage

```
design.full(x, par.constrained, constraints)
```

Arguments

x	A 2-column matrix of numeric values representing genotypes of biallelic markers, with one column per marker and one row per subject. The two homozygous genotypes must be coded 0 and 1, and the heterozygous genotype value depends on the genetic model: 0 (recessive), 1/2 (allelic) or 1 (dominant).
par.constrained	Optional matrix of dimensions (n.levels-1) x nc specifying the parameter (1, 2 or 3) in the linear predictor for each level involved in the nc constraints BETWEEN the logistic models for different levels of the response variable, one constraint per column. This functionality is not yet implemented.
constraints	Optional matrix of dimensions (n.levels-1) x nc specifying the nc linear constraints BETWEEN the logistic models for different levels of the response variable, involving the parameters specified in par.constrained, one constraint per column. A 0 means that the corresponding parameter is not involved in the constraint. This functionality is not yet implemented.

Details

Let Y_1 and Y_2 be binary variables coding the presence (1) or absence (0) of the two traits (e.g. and endophenotype and a disease trait, respectively). The linear predictors (without intercept) of the logistic functions between outcome levels and the reference level $Y_1 = 0$ and $Y_2 = 0$ for the full model are as follows:

$Y_1 = 1, Y_2 = 0 : \beta_{11}X_1 + \beta_{12}X_2 + \beta_{13}X_1X_2$

$Y_1 = 0, Y_2 = 1 : \beta_{21}X_1 + \beta_{22}X_2 + \beta_{23}X_1X_2$

$Y_1 = 1, Y_2 = 1 : \beta_{31}X_1 + \beta_{32}X_2 + \beta_{33}X_1X_2$

The design matrices for the above model are constructed by this function.

Value

x.e	List of 3 design matrices containing all terms forming the full model
x.loc.e	list of character strings containing the indices of the variables in x involved in each term of the full model
x.l	identical to x.e
x.loc.l	identical to x.l

Author(s)

Alexandre Bureau <alexandre.bureau@msp.ulaval.ca>

References

Bureau A., Croteau J., Chagnon, Y.C., Roy, M.-A. and Maziade, M. Extension of the Generalized Disequilibrium Test to polytomous phenotypes and two locus models. Frontiers in Genetics, 5: Article 258.

See Also

fat2Lpoly,design.endo2disease

---

fat2Lpoly *Two-locus Family-based Association Test with Polytomous Outcome*

Description

Performs family-based association tests with a polytomous outcome under 2-locus and 1-locus models as described in reference [1]. Various functions design.constraint to create design matrices are provided in this package. When SNP pairs are specified, the tested SNP is the second one of each pair, while the first one is considered the conditioning SNP. The function may also perform one-locus tests if individual SNPs are specified instead of SNP pairs.

Usage

```
fat2Lpoly(pedfilenames, datfilenames, freq.data, ibdfilenames = NULL,
          snp.names.mat, ibd.loci = NULL, joint.tests = NULL,
  contingency.file = FALSE, design.constraint,
  par.constrained, constraints, pairweights=calcule.poids.alphafixe,
  lc = NULL, alpha = NULL)
```

Arguments

pedfilenames	vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the pedigree files in Merlin format (see Merlin website [2]). Put the full path of the files if they are not in the current working directory. If the phenotype is polytomous with 4 levels created by all combinations of two dichotomous phenotypic variables Y_1 and Y_2, then the sixth and seventh columns of each file are respectively for Y_1 (e.g. the endophenotype) and Y_2 (e.g. the disease phenotype).
datfilenames	vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the Merlin format data files corresponding to the pedigree files.
freq.data	Either (1) a vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the allele frequency files corresponding to the pedigree files. These files must be in Merlin Classic format. or (2) a list of length 1 or 2 (the number of loci involved in the design function), each element of which is a numeric vector of length 'number of SNPs in datfilenames' and specifies each SNP's minor allele.
ibdfilenames	vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the Merlin format ibd files corresponding to the pedigree files. If NULL (the default), then we use the kinship coefficients multiplied by two, instead of the expectation of the IBD probabilities, in the computation of the score statistics. The kinship coefficients are obtained using the function kinship from the package kinship2.

`snp.names.mat` matrix of one or two columns giving the names of the SNPs (if one column matrix) or pairs of SNPs (if two columns matrix) to be analyzed. These SNPs represent all or part of the SNPs in the data files `datfilenames`.

`ibd.loci` matrix of the same dimensions as `snp.names.mat`, giving the respective names of the markers (used to obtain the IBD results) closest to the corresponding SNPs. The marker names must be written exactly the same as in the ibd files `ibdfilenames` for extraction of IBD data. If the IBD data are specified by genetic positions instead of marker names, then for each SNP, specify the genetic position where IBD was inferred which is closest to the corresponding SNP. If NULL (the default), then we use the kinship coefficients multiplied by two, instead of the expectation of the IBD probabilities, in the computation of the score statistics. The kinship coefficients are obtained using the function `kinship` from the package `kinship2`.

`joint.tests` list of vectors of numbers between 1 and the total number of parameters in the `design` function. Each vector gives parameter indices to test the corresponding parameters jointly. The default is not to perform any joint test.

`contingency.file`
if 'TRUE' (default is 'FALSE'), then a file called descriptive_statistics'date_and_time'.txt is created and contingency tables with the numbers of subjects per level are progressively added to this file.

`design.constraint`
function building the design matrices WITHIN each category, for constraints specific to each category. It also returns the design matrices comprising only the loci main effects that are used for computing the covariances. An attribute `n.levels` must be added within the function, to the object it returns.

`par.constrained`
Optional matrix of dimensions (`n.levels`-1) x nc specifying the parameter in the linear predictor for each level involved in the nc constraints BETWEEN the logistic models for different levels of the response variable, one constraint per column. This functionality is not yet implemented.

`constraints` Optional matrix of dimensions (`n.levels`-1) x nc specifying the nc linear constraints BETWEEN the logistic models for different levels of the response variable, involving the parameters specified in `par.constrained`, one constraint per column. A 0 means that the corresponding parameter is not involved in the constraint. This functionality is not yet implemented.

`pairweights` function calculating the weights of the observation pair differences when conditioning on the first SNP in the test of the second SNP in a SNP pair. Default is calcule.poids.alphafixe, implementing the weighting function of equation (6) of reference [1]. An alternative is calcule.poids.Chen, implementing the weighting function of equation (7) of reference [1].

`lc` numerical identifier of the SNP (locus) on which to condition when testing model terms. Defaults to NULL, or no conditioning.

`alpha` vector of length `n.levels`-1 of the coefficients of the polytomous logistic model of association beween the phenotype and the conditionning SNP. Defaults to NULL. If `alpha` = `NULL` and `lc` is not NULL, an `alpha` is obtained by logistic regression (multinomial logistic regression if `n.levels`>2) of the phenotype on the genotype at locus `lc`.

Details

All subjects included in the pedigree files must also be found in the IBD files.

All fields in the pedigree files must be numeric. No letters allowed, even for family and subject ID's.

Families whose genotyped subjects are all in the same category (phenotype combination), are uninformative and will be excluded.

Conditioning on the first SNP in a SNP pair is implemented by weighting the observation pair differences according to a model of the polytomous outcome as a function of the first SNP genotypes. The function converting the coefficients of this regression model into weights is specified by the argument `pairweights`. The default function `calcule.poids.alphafixe` provided satisfactory power in simulations described in reference [1].

File "descriptive_statistics'date_and_time'.txt" (will be created if contingency.file='TRUE'): For each tested SNP, it shows contingency tables of the subjects in the 2 or 4 different categories, first for all families together and then for each individual family.

If one or both of the arguments ibd.loci and ibdfilenames are left unspecified (or NULL, their default), then we use the kinship coefficients multiplied by two, instead of the expectation of the IBD probabilities, in the computation of the score statistics. The kinship coefficients are obtained using the function `kinship` from the package `kinship2`.

Value

returns a list of five objects:

`scores.covs.all.SNPs`
: list of length 'nrow(`snp.names.mat`)', each element of which contains the estimates of the scores and covariances of all the families.

`p.values.scores`
: data frame of p-values for all the SNPs or SNP pairs in `snp.names.mat`, for the global test (all parameters tested jointly), the individual tests and other joint tests specified by the argument `joint.tests`. The p-values are obtained from scores summed over all families. These scores of individual tests are also included in this data frame.

`MA.table`
: data frame giving the minor allele numbers of all the SNPs contained in the allele frequency files.

`y1`
: affection name extracted from first line of the data file(s)

`y2`
: affection name extracted from second line of the data file(s)

Author(s)

Alexandre Bureau and Jordie Croteau

References

1. Bureau A., Croteau J., Chagnon, Y.C., Roy, M.-A. and Maziade, M. Extension of the Generalized Disequilibrium Test to polytomous phenotypes and two locus models. Frontiers in Genetics, 5: Article 258. 2. http://www.sph.umich.edu/csg/abecasis/Merlin/tour/input_files.html

See Also

fat2Lpoly.withinR

Examples

```
path.data=paste(.libPaths()[which(unlist(lapply(.libPaths(),
function(x) length(grep("fat2Lpoly",dir(x)))))>0)],"/fat2Lpoly/extdata/",sep="")
if(length(path.data)>1) path.data=path.data[length(path.data)]

snps.anal=c("snp3.loc2","snp4.loc2")
microsat.names.loc2=c("2_3_mrk:","2_4_mrk:")

############ design.endo2disease with conditioning on locus 1 ################
## Not run:
joint.tests=list(c(2,5))
snp.names.mat=cbind(rep("snp4.loc1",length(snps.anal)),snps.anal)
microsat.names.mat=cbind(rep("1_4_mrk:",length(snps.anal)),microsat.names.loc2)
test=fat2Lpoly(pedfilenames=paste(path.data,c("loc1.ped","loc2.ped"),sep=""),
               datfilenames=paste(path.data,c("loc1.dat","loc2.dat"),sep=""),
   freq.data=paste(path.data,c("loc1.freq","loc2.freq"),sep=""),
               ibdfilenames=paste(path.data,c("loc1.ibd","loc2.ibd"),sep=""),
       snp.names.mat=snp.names.mat,ibd.loci=microsat.names.mat,
       joint.tests=joint.tests,contingency.file=TRUE,
       design.constraint=design.endo2disease,lc=1)

test$p.values.scores

## End(Not run)
###############################################################################

################### design.endo2disease without conditioning #################
joint.tests=list(c(2,5))
snp.names.mat=cbind(rep("snp4.loc1",length(snps.anal)),snps.anal)
microsat.names.mat=cbind(rep("1_4_mrk:",length(snps.anal)),microsat.names.loc2)
test=fat2Lpoly(pedfilenames=paste(path.data,c("loc1.ped","loc2.ped"),sep=""),
               datfilenames=paste(path.data,c("loc1.dat","loc2.dat"),sep=""),
   freq.data=paste(path.data,c("loc1.freq","loc2.freq"),sep=""),
               ibdfilenames=paste(path.data,c("loc1.ibd","loc2.ibd"),sep=""),
       snp.names.mat=snp.names.mat,ibd.loci=microsat.names.mat,
       joint.tests=joint.tests,contingency.file=FALSE,
       design.constraint=design.endo2disease)

test$p.values.scores
###############################################################################

################ design.full with conditioning on locus 1 #################
## Not run:
joint.tests=list(c(2,3),c(5,6),c(8,9),c(2,3,5,6,8,9))
snp.names.mat=cbind(rep("snp4.loc1",length(snps.anal)),snps.anal)
microsat.names.mat=cbind(rep("1_4_mrk:",length(snps.anal)),microsat.names.loc2)
test=fat2Lpoly(pedfilenames=paste(path.data,c("loc1.ped","loc2.ped"),sep=""),
               datfilenames=paste(path.data,c("loc1.dat","loc2.dat"),sep=""),
```

```
   freq.data=paste(path.data,c("loc1.freq","loc2.freq"),sep=""),
               ibdfilenames=paste(path.data,c("loc1.ibd","loc2.ibd"),sep=""),
       snp.names.mat=snp.names.mat,ibd.loci=microsat.names.mat,
       joint.tests=joint.tests,
               design.constraint=design.full,lc=1)

test$p.values.scores

## End(Not run)
################################################################################

############################## design.1locus ##################################
snp.names.mat=as.matrix(snps.anal)
microsat.names.mat=as.matrix(microsat.names.loc2)
test=fat2Lpoly(pedfilenames=paste(path.data,"loc2.ped",sep=""),
               datfilenames=paste(path.data,"loc2.dat",sep=""),
               freq.data=paste(path.data,"loc2.freq",sep=""),
   ibdfilenames=paste(path.data,"loc2.ibd",sep=""),
       snp.names.mat=snp.names.mat,ibd.loci=microsat.names.mat,
   design.constraint=design.1locus)

test$p.values.scores
################################################################################

############# design.dichotomous with conditioning on locus 1 ##############
## Not run:
joint.tests=list(c(2,3))
snp.names.mat=cbind(rep("snp4.loc1",length(snps.anal)),snps.anal)
microsat.names.mat=cbind(rep("1_4_mrk:",length(snps.anal)),microsat.names.loc2)
test=fat2Lpoly(pedfilenames=paste(path.data,c("loc1.ped","loc2.ped"),sep=""),
               datfilenames=paste(path.data,c("loc1.dat","loc2.dat"),sep=""),
   freq.data=paste(path.data,c("loc1.freq","loc2.freq"),sep=""),
               ibdfilenames=paste(path.data,c("loc1.ibd","loc2.ibd"),sep=""),
       snp.names.mat=snp.names.mat,ibd.loci=microsat.names.mat,
       joint.tests=joint.tests,
               design.constraint=design.dichotomous,lc=1)

test$p.values.scores

## End(Not run)
################################################################################
```

`fat2Lpoly.allSNPs` *Example results output by the function* `fat2Lpoly.withinR`

Description

This list is an example of output from the function `fat2Lpoly.withinR`. It is provided to test the function `get.scores.pvalues` by executing the example code in the `get.scores.pvalues` documentation.

Usage

```
data(fat2Lpoly.allSNPs)
```

Format

A list of two objects:

scores.covs.all.SNPs list of length 'nrow(snp.names.mat)', each element of which contains the estimates of the scores and covariances of all the families.

snp.names.mat (same matrix as provided as argument) matrix of one or two columns giving the names of the SNPs (if one column matrix) or pairs of SNPs (if two columns matrix) to be analyzed. These SNPs represent all or part of the SNPs in the data files datfilenames.

Examples

```
data(fat2Lpoly.allSNPs)
```

---

fat2Lpoly.withinR *Two-locus Family-based Association Test with Polytomous Outcome (all arguments within R)*

Description

Same as fat2Lpoly except that the first four arguments of fat2Lpoly are replaced by one object having the format of the objects returned by read.merlin.files.

Usage

```
fat2Lpoly.withinR(ped.x.all, snp.names.mat, ibd.loci = NULL, contingency.file = FALSE,
                  design.constraint, par.constrained, constraints,
                  pairweights=calcule.poids.alphafixe, lc = NULL, alpha = NULL)
```

Arguments

ped.x.all	object returned by the function read.merlin.files or having the same format.
snp.names.mat	matrix of one or two columns giving the names of the SNPs (if one column matrix) or pairs of SNPs (if two columns matrix) to be analyzed. These SNPs represent all or part of the SNPs in the data files datfilenames.
ibd.loci	matrix of the same dimensions as snp.names.mat, giving the respective names of the markers (used to obtain the IBD results) nearest to the corresponding SNPs. The marker names must be written exactly the same as in the ibd files ibdfilenames for extraction of IBD data. If the IBD data are specified by genetic positions instead of marker names, this matrix must contain the genetic positions of the markers instead of the marker names. If NULL (the default), then we use the kinship coefficients multiplied by two, instead of the expectation of the IBD probabilities, in the computation of the score statistics. The kinship coefficients are obtained using the function kinship from the package kinship2.

`contingency.file`
: if 'TRUE' (default is 'FALSE'), then a file called descriptive_statistics'date_and_time'.txt is created and contingency tables with the numbers of subjects per level are progressively added to this file.

`design.constraint`
: function building the design matrices WITHIN each category, for constraints specific to each category. It also returns the design matrices comprising only the loci main effects that are used for computing the covariances.

`par.constrained`
: Optional matrix of dimensions (`n.levels`-1) x nc specifying the parameter in the linear predictor for each level involved in the nc constraints BETWEEN the logistic models for different levels of the response variable, one constraint per column. This functionality is not yet implemented.

`constraints`
: Optional matrix of dimensions (`n.levels`-1) x nc specifying the nc linear constraints BETWEEN the logistic models for different levels of the response variable, involving the parameters specified in `par.constrained`, one constraint per column. A 0 means that the corresponding parameter is not involved in the constraint. This functionality is not yet implemented.

`pairweights`
: function calculating the weights of the observation pair differences when conditioning on the first SNP in the test of the second SNP in a SNP pair. Default is calcule.poids.alphafixe, implementing the weighting function of equation (6) of reference [1]. An alternative is calcule.poids.Chen, implementing the weighting function of equation (7) of reference [1].

`lc`
: numerical identifier of the SNP (locus) on which to condition when testing model terms. Defaults to NULL, or no conditioning.

`alpha`
: vector of length `n.levels`-1 of the coefficients of the polytomous logistic model of association beween the phenotype and the conditionning SNP. Defaults to NULL. If `alpha` = `NULL` and `lc` is not NULL, an `alpha` is obtained by logistic regression (multinomial logistic regression if `n.levels`>2) of the phenotype on the genotype at locus `lc`.

Details

File "descriptive_statistics'date_and_time'.txt" (will be created if contingency.file='TRUE'): For each tested SNP, it shows contingency tables of the subjects in the 2 or 4 different categories, first for all families together and then for each individual family.

If the argument ibd.loci is left unspecified (or NULL, its default), then we use the kinship coefficients multiplied by two, instead of the expectation of the IBD probabilities, in the computation of the score statistics. The kinship coefficients are obtained using the function `kinship` from the package `kinship2`.

Value

`scores.covs.all.SNPs`
: list of length 'nrow(`snp.names.mat`)', each element of which contains the estimates of the scores and covariances of all the families.

`snp.names.mat` (same matrix as provided as argument) matrix of one or two columns giving the names of the SNPs (if one column matrix) or pairs of SNPs (if two columns matrix) to be analyzed. These SNPs represent all or part of the SNPs in the data files `datfilenames`.

Author(s)

Alexandre Bureau and Jordie Croteau

References

Bureau A., Croteau J., Chagnon, Y.C., Roy, M.-A. and Maziade, M. Extension of the Generalized Disequilibrium Test to polytomous phenotypes and two locus models. Frontiers in Genetics, 5: Article 258.

See Also

`fat2Lpoly`,`read.merlin.files`,`get.scores.pvalues`

Examples

```
data(ped.x.all)

## Not run:
snp.names.mat=cbind(rep("snp4.loc1",2),c("snp3.loc2","snp4.loc2"))
microsat.names.mat=cbind(rep("1_4_mrk:",2),c("2_3_mrk:","2_4_mrk:"))
fat2Lpoly.allSNPs=fat2Lpoly.withinR(ped.x.all,snp.names.mat,ibd.loci=
                            microsat.names.mat,contingency.file=TRUE,
design.constraint=design.endo2disease,
lc=1)

joint.tests=list(c(2,5))
get.scores.pvalues(fat2Lpoly.allSNPs,joint.tests)

## End(Not run)
```

`get.scores.pvalues` *function to compute scores and p-values*

Description

For each tested SNP and each parameter in the model, computes scores by summing family scores over all families and computes the corresponding p-values. P-values of global and joint tests are also computed.

Usage

```
get.scores.pvalues(test, joint.tests)
```

Arguments

test	object returned by fat2Lpoly.withinR.
joint.tests	list of vectors of numbers between 1 and the total number of parameters in the design function. Each vector gives parameter indices to test the corresponding parameters jointly.

Value

data frame of p-values for all the tested SNPs, for the global test (all parameters tested jointly), the individual tests and other joint tests specified by the argument joint.tests. The p-values are obtained from scores summed over all families. These scores of individual tests are also included in this data frame.

Author(s)

Alexandre Bureau and Jordie Croteau

See Also

fat2Lpoly, fat2Lpoly.withinR

Examples

```
data(fat2Lpoly.allSNPs)

joint.tests=list(c(2,5),c(3,4))

get.scores.pvalues(fat2Lpoly.allSNPs, joint.tests)

#    snp.cond  snp.test global_p params.joint_2-5_p params.joint_3-4_p param_1_score
# 1 snp4.loc1 snp2.loc2 5.80e-03           7.12e-01           0.000954         0.449
# 2 snp4.loc1 snp4.loc2 2.14e-07           1.24e-05           0.000954         0.449
# 3 snp4.loc1 snp5.loc2 1.14e-03           1.44e-01           0.000954         0.449
# 4 snp4.loc1 snp6.loc2 5.59e-04           3.84e-02           0.000954         0.449
# 5 snp4.loc1 snp8.loc2 1.15e-03           1.55e-01           0.000954         0.449
# param_2_score param_3_score param_4_score param_5_score param_1_p param_2_p
#         0.333        -1.427         3.638         0.733     0.653     0.739
#         0.890        -1.427         3.638         4.612     0.653     0.373
#         0.776        -1.427         3.638         1.785     0.653     0.438
#        -0.082        -1.427         3.638         2.553     0.653     0.934
#         0.869        -1.427         3.638         1.695     0.653     0.385
#   param_3_p param_4_p param_5_p
# 1     0.154  0.000275  0.464000
# 2     0.154  0.000275  0.000004
# 3     0.154  0.000275  0.074200
# 4     0.154  0.000275  0.010700
# 5     0.154  0.000275  0.090100
```

ped.x.all *Example dataset returned by the function* read.merlin.files

Description

This list is an example of output from the function read.merlin.files. It is provided to test the function fat2Lpoly.withinR by executing the example code in the fat2Lpoly.withinR documentation.

Usage

```
data(ped.x.all)
```

Format

A list of six objects:

ped data frame with columns fam.id, subject.ids, endophenotype and phenotype (in the given order)

x.all data frame of SNP genotypes in the format "(number of minor alleles)/2", for all SNPs listed in the file(s) in datfilenames. It contains only the SNP data and it has as column names the SNP names in datfilenames. The lines come in the same order as in ped.

MA.table data frame giving the minor allele numbers of all the SNPs. The first column consists of x.all's column names and the second column the minor allele numbers.

ibd.dat.list list of one or two data frames containing the columns of the IBD data file(s) in ibdfilenames.

y1.name affection name extracted from first line of the data file(s)

y2.name affection name extracted from second line of the data file(s)

ibdfilenames (same object as provided as argument) vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the Merlin format ibd files corresponding to the pedigree files.

Examples

```
data(ped.x.all)
```

read.merlin.files *function to read input files in Merlin format*

Description

Reads the pedigree, data and allele frequency input files. The data read is reformatted to be used by the function fat2Lpoly.withinR.

Usage

```
read.merlin.files(pedfilenames, datfilenames, freq.data, ibdfilenames = NULL)
```

Arguments

pedfilenames	vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the pedigree files in Merlin format (see Merlin website [1]). Put the full path of the files if they are not in the current working directory. If the phenotype is polytomous with 4 levels created by all combinations of two dichotomous phenotypic variables Y_1 and Y_2, then the sixth and seventh columns of each file contain respectively Y_1 (e.g. the endophenotype) and Y_2 (e.g. the disease phenotype). If the phenotype is dichotomous, then the sixth column of each file contains it.
datfilenames	vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the Merlin format data files corresponding to the pedigree files.
freq.data	Either (1) a vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the allele frequency files corresponding to the pedigree files. These files must be in Merlin Classic format. or (2) a list of length 1 or 2 (the number of loci involved in the design function), each element of which is a numeric vector of length 'number of SNPs in datfilenames' and specifies each SNP's minor allele.
ibdfilenames	vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the Merlin format ibd files corresponding to the pedigree files. If NULL (the default), the reading of the IBD files is skipped.

Details

All subjects included in the pedigree files must also be found in the IBD files.

All fields in the pedigree files must be numeric. No letters allowed, even for family and subject ID's.

Value

returns a list of six objects:

ped	data frame with columns fam.id, subject.ids, endophenotype and phenotype (in the given order)
x.all	data frame of SNP genotypes in the format "(number of minor alleles)/2", for all SNPs listed in the file(s) in datfilenames. It contains only the SNP data and it has as column names the SNP names in datfilenames. The lines come in the same order as in ped.
MA.table	data frame giving the minor allele numbers of all the SNPs. The first column consists of x.all's column names and the second column the minor allele numbers.
ibd.dat.list	list of one or two data frames containing the columns of the IBD data file(s) in ibdfilenames.

y1.name	affection name extracted from first line of the data file(s)
y2.name	affection name extracted from second line of the data file(s)
ibdfilenames	(same object as provided as argument) vector of 1 or 2 (the number of loci involved in the design function) character strings giving the names of the Merlin format ibd files corresponding to the pedigree files.

Author(s)

Alexandre Bureau and Jordie Croteau

References

1. http://www.sph.umich.edu/csg/abecasis/Merlin/tour/input_files.html

See Also

fat2Lpoly.withinR

Examples

```
path.data=paste(.libPaths()[which(unlist(lapply(.libPaths(),
function(x) length(grep("fat2Lpoly",dir(x))))>0)],
"/fat2Lpoly/extdata/",sep="")
if(length(path.data)>1) path.data=path.data[length(path.data)]

input.data=read.merlin.files(pedfilenames=
                  paste(path.data,c("loc1.ped","loc2.ped"),sep=""),
                  datfilenames=
paste(path.data,c("loc1.dat","loc2.dat"),sep=""),
      freq.data=
      paste(path.data,c("loc1.freq","loc2.freq"),sep=""),
                        ibdfilenames=
paste(path.data,c("loc1.ibd","loc2.ibd"),sep=""))

input.data2=read.merlin.files(pedfilenames=
paste(path.data,"loc2.ped",sep=""),
                             datfilenames=
                             paste(path.data,"loc2.dat",sep=""),
                             freq.data=
                             paste(path.data,"loc2.freq",sep=""),
ibdfilenames=
paste(path.data,"loc2.ibd",sep=""))
```

Index

∗Topic **datasets**
fat2Lpoly.allSNPs, 11
ped.x.all, 16

design.1locus, 2
design.dichotomous, 3
design.endo2disease, *3*, 4, *7*
design.full, *3–5*, 5

fat2Lpoly, *3–5*, *7*, 7, *14, 15*
fat2Lpoly.allSNPs, 11
fat2Lpoly.withinR, *10*, 12, *15*, *18*

get.scores.pvalues, *14*, 14

kinship, *7–9*, *12, 13*

ped.x.all, 16

read.merlin.files, *14*, 16